



Desenvolvimento de Jogo do Gênero Puzzle



Continuando a construção de nosso jogo, iremos acrescentar agora os objetos quebráveis, inimigos e sons:

- **Criação de objeto quebrável:** aprender a fazer a muralha quebrar;
- **Criação inimigos:** aprender a fazer a mecânica do personagem receber visualmente dano e morrer;
- **Áudios Isolados Personagens:** saber colocar sons nos personagens;
- **Áudio para objetos:** colocar sons nos objetos que só emitirão quando atingido pelo espartano.

- **Criação de objeto quebrável**

Iremos fazer agora a muralha de uma forma que ela vá se danificando de acordo com os impactos dos personagens nela. Para tanto será necessário que em um software de computação gráfica vetorial, você ilustre pelo men[illegible] 3 etapas da destruição do objeto (Figura 1).





Figura 1 – Muralha com fases de destruição | Fonte: Autor, 2022

Com isso iremos criar o código C#, nomeando-o como ImpactCol (impacto de colisão) para que a ação funcione, para tanto você precisará criar as variáveis:

- private int limite;
- private SpriteRenderer spriteR;
- [SerializeField]
- private Sprite[] sprites;

No Start você criará os objetos:

- limite = 0;
- spriteR = GetComponent<SpriteRenderer>();
- spriteR.sprite = sprites[0];

Depois disso é só criar o método para que o objeto se torne quebrável (Figura 2). Nesse caso fiz um código simples que irá destruir o objeto quando chegar no final do método, porém, você pode incrementar com partículas como fizemos com a morte do personagem espartano.

```
{
    if (col.relativeVelocity.magnitude > 2 && col.relativeVelocity.magnitude < 100)
    {
        if (limite < sprites.Length - 1)
        {
            limite++;
            spriteR.sprite = sprites[limite];
        }
        else if (limite == sprites.Length -1)
        {
            Destroy(gameObject);
        }
    }
    else if (col.relativeVelocity.magnitude > 12 && col.gameObject.CompareTag ("Player"))
    {
        Destroy(gameObject);
    }
}
```

Figura 2 – Código para quebra de objeto | Fonte: adaptado de NASCIMENTO, 2022.

Após isso é a hora de você lincar os objetos no “Inspector” no Unity (Figura 3).

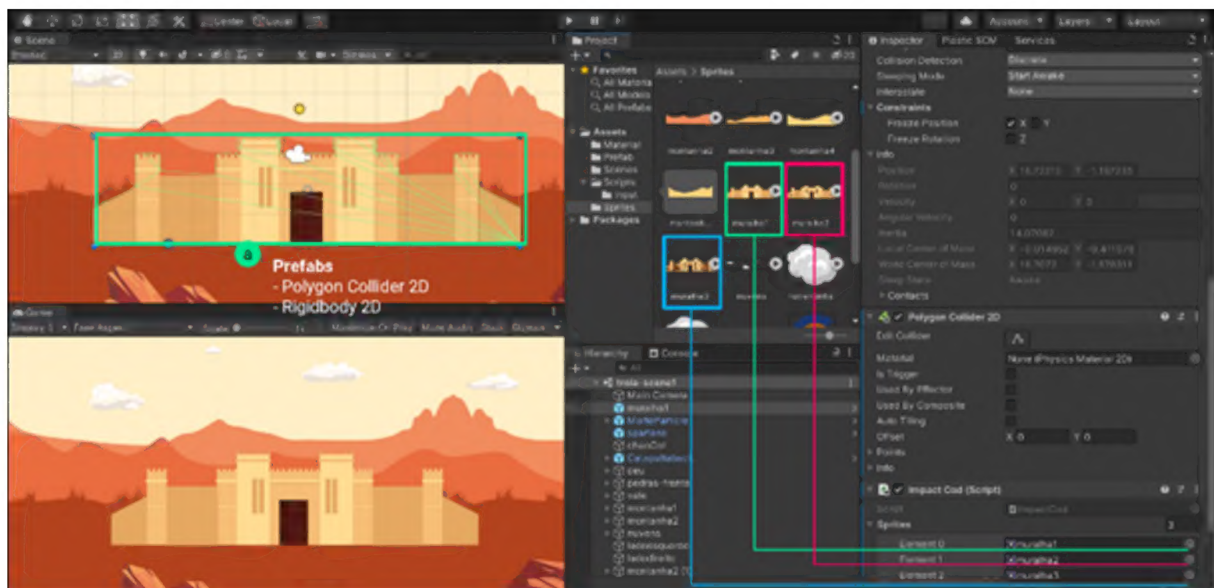


Figura 3 – Lincando objetos no Inspector | Fonte: Autor, 2022.



2. **Criando inimigos**

Para criar os inimigos, como o exemplo dos porcos em Angry Birds, iremos criar sprites para representar os troianos (Figura 4). Desenhei três variações com níveis de danos diferentes.

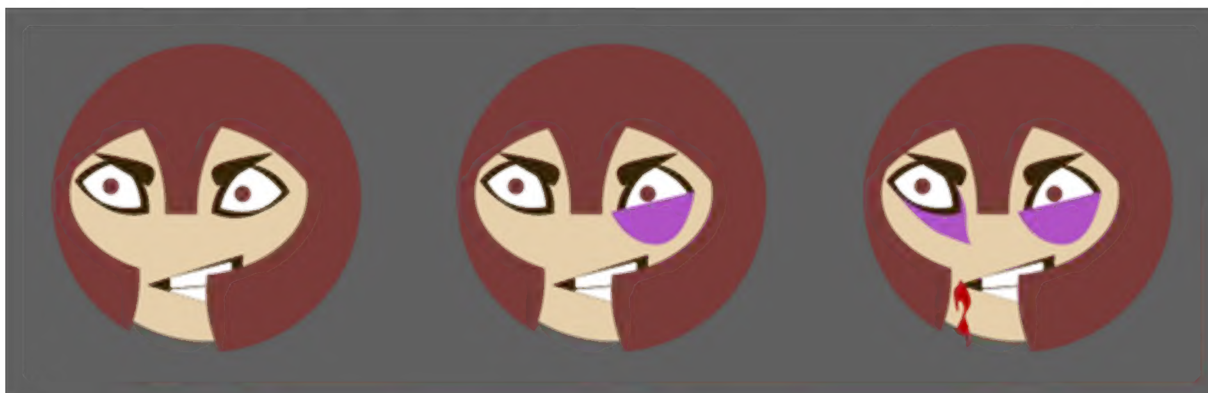


Figura 4 – Figura dos Troianos | Fonte: Autor, 2022.

O mesmo código C# que você criou para a muralha, poderá ser utilizado para gerar a ação com os personagens. O mesmo processo deverá ser adotado para que cada pancada seja sentida pelo objeto (Figura 5).

Porém, nesse código eu acrescentei duas coisas, um efeito de partículas e a pontuação de 1000 pt para a destruição do personagem (Figura 6).

Depois é só usar o "Ctrl+D" para duplicar os personagens.

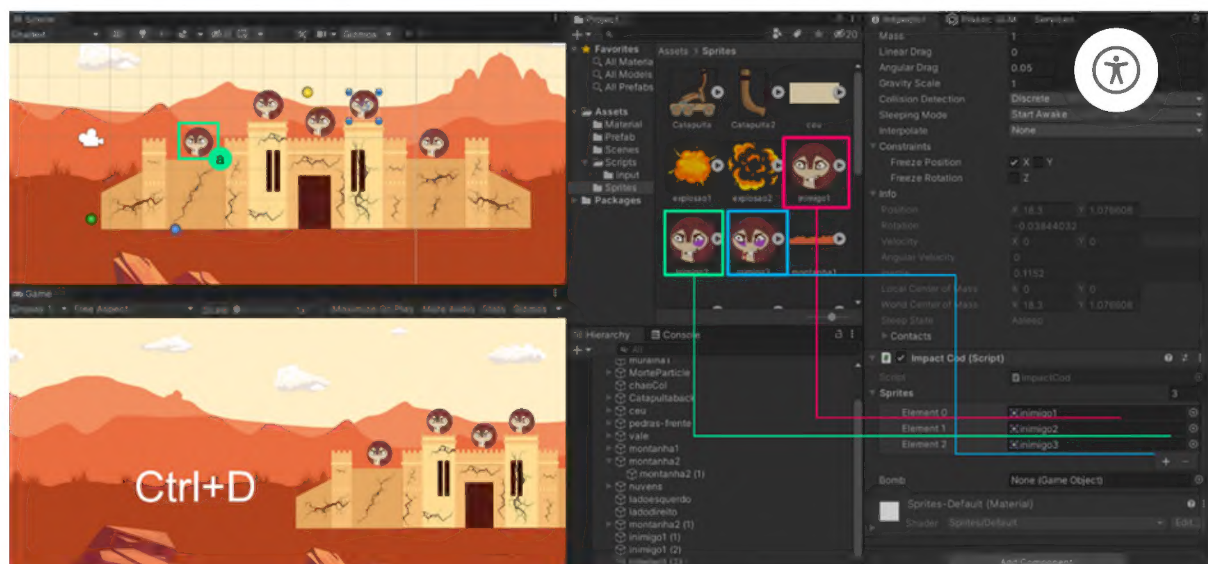


Figura 5 – Código C# da Muralha sendo usado nos Troianos | Fonte: Autor, 2022.

```
    else if (limite == sprites.Length -1)
    {
        Instantiate(pontos1000, new Vector2(transform.position.x, transform.position.y), Quaternion.identity);
        Instantiate(bomb, new Vector2(transform.position.x, transform.position.y), Quaternion.identity);
        Destroy(gameObject);
    }
}
else if (col.relativeVelocity.magnitude > 12 && col.gameObject.CompareTag ("Player"))
{
    Instantiate(pontos1000, new Vector2(transform.position.x, transform.position.y), Quaternion.identity);
    Instantiate(bomb, new Vector2(transform.position.x, transform.position.y), Quaternion.identity);
    Destroy(gameObject);
}
```

Figura 6 – Incremento de código C# | Fonte: Autor, 2022.

3. **Áudios Isolados Personagens**

Para acrescentar um áudio isolado nos personagens que representam os inimigos é só selecionar o troiano, ir em "Components", "Áudio" e por fim em "Audio Source".

Para os seus efeitos sonoros, você pode buscar várias opções na internet, porém, recomendo que busque áudios livres de direito de uso ou que você possa criar os seus próprios.

Para esse trabalho eu utilizei a biblioteca de áudio do Youtube. O YouTube oferece a todos os usuários da plataforma uma nova biblioteca de áudio. São 150 trilhas sonoras gratuitas e livres de direitos autorais. O mais interessante

dessa ferramenta é que você pode escolher a música por gênero, clima, duração etc.

Depois que você colocar o “Audio Source” é só linkar o áudio com os Prefebs dos inimigos (Figura 7).

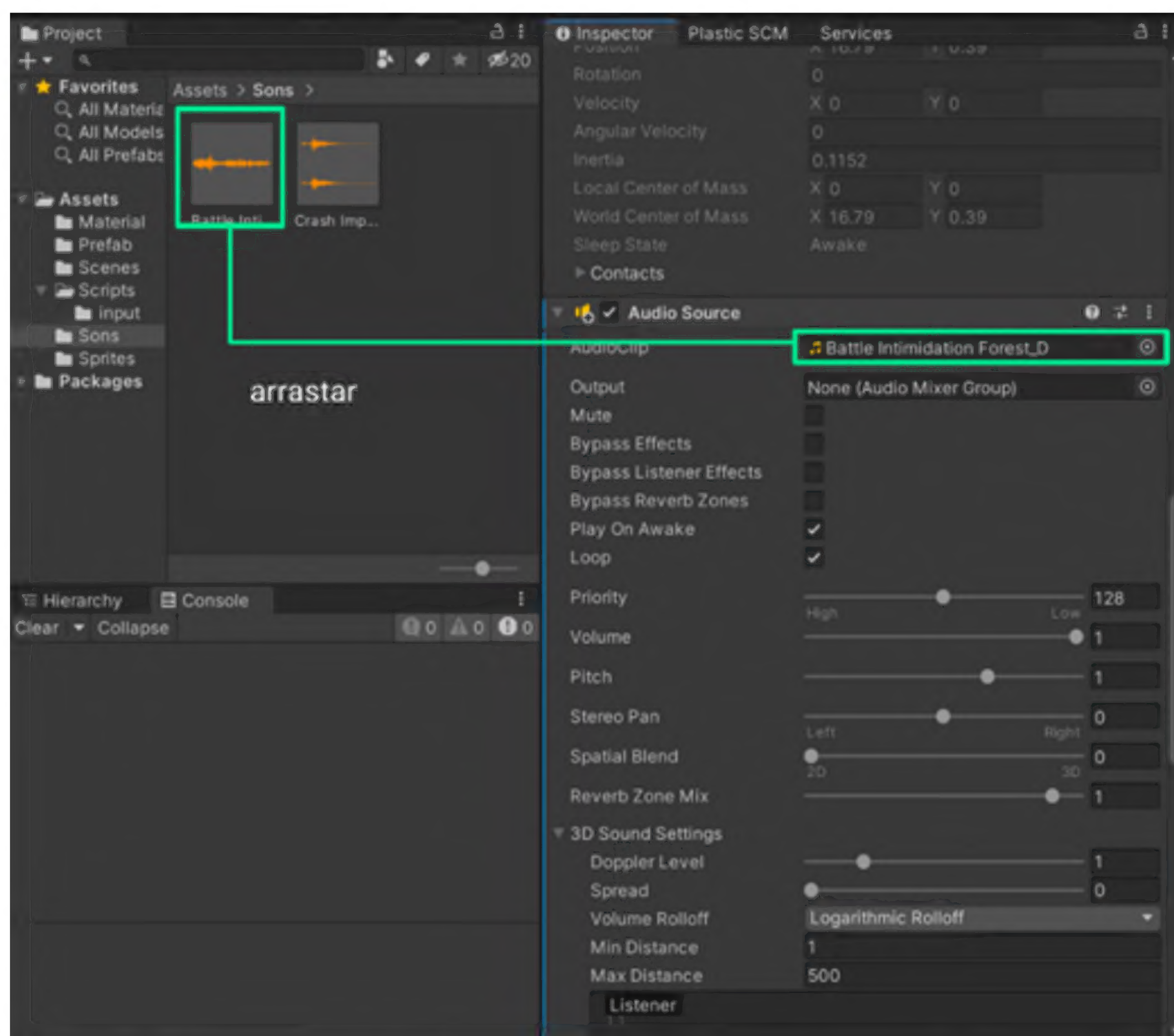


Figura 7 – Áudios isolados inimigos | Fonte: Autor, 2022.

Agora vamos colocar a parte de som no espartano. Para tanto iremos dessa vez usar um código, pois espero que o som saia apenas quando ele for lançado.

Para tanto, selecione o Espartano, vá em “Components”, “Audio” e por fim clique em “Audio Source”. Acrescente o áudio que quiser, depois é a hora de codificar em “Drag”. Agora no código C# você precisará:

- Acrescentar a variante “public AudioSource audioEspartano;”;

- Em Start, colocar o objeto “audioEspartano = GetComponent<AudioSource>();”;

- No código onde se coloca para soltar o pássaro, acrescentar um play “audioEspartano.Play();”.

4. **Áudio para objetos**

O início é muito parecido com os anteriores, ou seja, você precisará selecionar o objeto, ir em “Components”, “Audio” e por fim clicar em “Audio Source”. Em seguida deverá inserir o som escolhido e codificar o arquivo de código “ImpactCod”.

No código você deverá colocar as seguintes variantes e código em Start:

Variáveis	Void Start
private AudioSource audioObj; [SerializeField] private AudioClip[] clips;	audioObj = GetComponent<AudioSource>();

Tabela 1 – Códigos C# som objetos | Fonte: Autor, 2022

Depois você precisará incluir os códigos na colisão e destruição do objeto (Figura 8).



```
// Mensagem do Unity | 0 referências
void OnCollisionEnter2D(Collision2D col)
{
    if (col.relativeVelocity.magnitude > 2 && col.relativeVelocity.magnitude < 100)
    {
        if (limite < sprites.Length - 1)
        {
            limite++;
            spriteR.sprite = sprites[limite];
            audioObj.clip = clips[0];
            audioObj.Play();
        }
        else if (limite == sprites.Length -1)
        {
            Instantiate(pontos1000, new Vector2(transform.position.x, transform.position.y), Quaternion.identity);
            Instantiate(bomb, new Vector2(transform.position.x, transform.position.y), Quaternion.identity);
            audioObj.clip = clips[1];
            audioObj.Play();
            Destroy(gameObject);
        }
    }
    else if (col.relativeVelocity.magnitude > 12 && col.gameObject.CompareTag ("Player"))
    {
        Instantiate(pontos1000, new Vector2(transform.position.x, transform.position.y), Quaternion.identity);
        Instantiate(bomb, new Vector2(transform.position.x, transform.position.y), Quaternion.identity);
        audioObj.clip = clips[1];
        audioObj.Play();
        Destroy(gameObject);
    }
}
```

Figura 8 – Código para áudios isolados dos objetos | Fonte: Autor, 2022.

Atividade Extra

Assista os vídeos sobre Áudio para Jogos! O produtor de áudio para jogos Thiago Adamo, que gravou uma mensagem para nossos estudantes da Faculdade Descomplica que passamos normalmente no início do curso, tem um canal fantástico no Youtube. Veja algumas dicas de como escolher a melhor trilha sonora.

https://www.youtube.com/c/pxldj/about

Referência Bibliográfica



NASCIMENTO, W. **Jogos 2D com Unity e C#**. São Paulo: Udemy, 2022.

SATO, A. K. O.; CARDOSO, M. V. **Além do gênero**: uma possibilidade para a classificação de jogos. SBC - Proceedings of SBGames'08: Art & Design Track. Belo Horizonte - MG, November 10. p. 54 - 63. Disponível em: <http://www.sbgames.org/papers/sbgames08/ad/papers/p08.pdf>.

Atividade Prática: Desenvolvimento de Jogo do Gênero Puzzle

Título da Prática: Desenvolvimento de Jogo do Gênero Puzzle

Objetivos: Criar objetos quebráveis com animações e som.

Materiais, Métodos e Ferramentas: Para realizar esta prática vamos utilizar o software Unity, um smartphone Android e acesso ao Youtube.

Atividade Prática

Roteiro



1º Passo – Leia o texto do material de apoio.

2º Passo – Assista aos vídeos da disciplina e aula 15.

3º Passo – Usando os passos a passos explicado e feito nos vídeos e Material de Apoio, criar a mecânica de objeto quebrado e sons.

4º Passo – Crie o efeito da muralha quebrando por meio do Código C# demonstrado em material de apoio.

5º Passo – Adicione ruídos de quebra ao seu objeto.

6º Passo – Repita isso ao seu inimigo.

7º Passo – Verificar se não existe erros e como corrigi-los.

8º Passo – Crie um relatório de tudo o que você fez, bem como erros que surgiram e como você os resolveu.

Após a conclusão da Atividade poste o relatório e responda:

1) Quais erros você teve que enfrentar durante o desenvolvimento da mecânica do seu jogo? Como resolveu?

2) O que e como aprendeu?



Gabarito Atividade Prática

1) Se você deixar de colocar símbolos como “;”, “()” ou “{}” pode ser que você tenha encontrado algum tipo de erro, ainda mais se a sintaxe do seu código estiver errada. Pode ser que ao digitar o nome do objeto você também tenha cometido algum erro. Porém, o próprio Unity mostra os erros e indica a linha do código onde ele se encontra. Dependendo do tipo de editor de código que você está usando, ele poderá de apresentar possibilidades de resolução de problemas.

2) Você aprendeu utilizando diversas técnicas e códigos. Criando a mecânica principal do jogador, ou seja, do player. Com isso você já teria um produto para apresentar em uma Game Jam. Aprendeu de forma ativa, criando códigos e conectando objetos a eles.

Ir para exercício